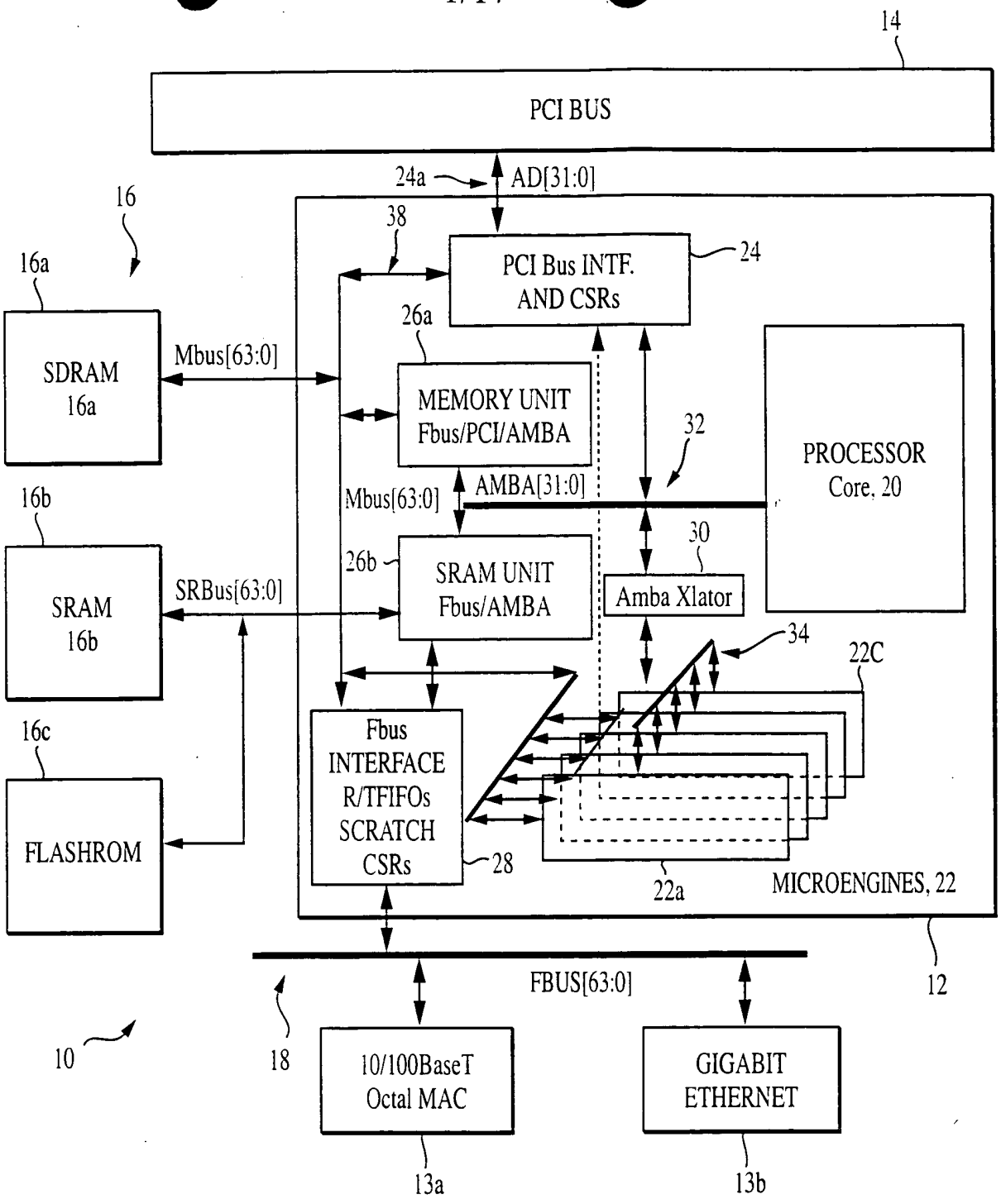

14
PCI BUS
24a
AD[31:0]
16
38
16a
PCI Bus INTF.
AND CSRs
24
26a
SDRAM
16a
Mbus[63:0]
MEMORY UNIT
Fbus/PCI/AMBA
32
PROCESSOR
Core, 20
Mbus[63:0]
AMBA[31:0]
16b
30
26b
SRAM
16b
SRBus[63:0]
SRAM UNIT
Fbus/AMBA
Amba Xlator
34
22C
16c
Fbus
INTERFACE
R/TFIFOs
SCRATCH
CSRs
28
FLASHROM
22a
MICROENGINES, 22
12
FBUS[63:0]
10
18
10/100BaseT
Octal MAC
GIGABIT
ETHERNET
13a
13b

FIG. 1

FIG. 2-1

FIG. 2-2

FIG. 2

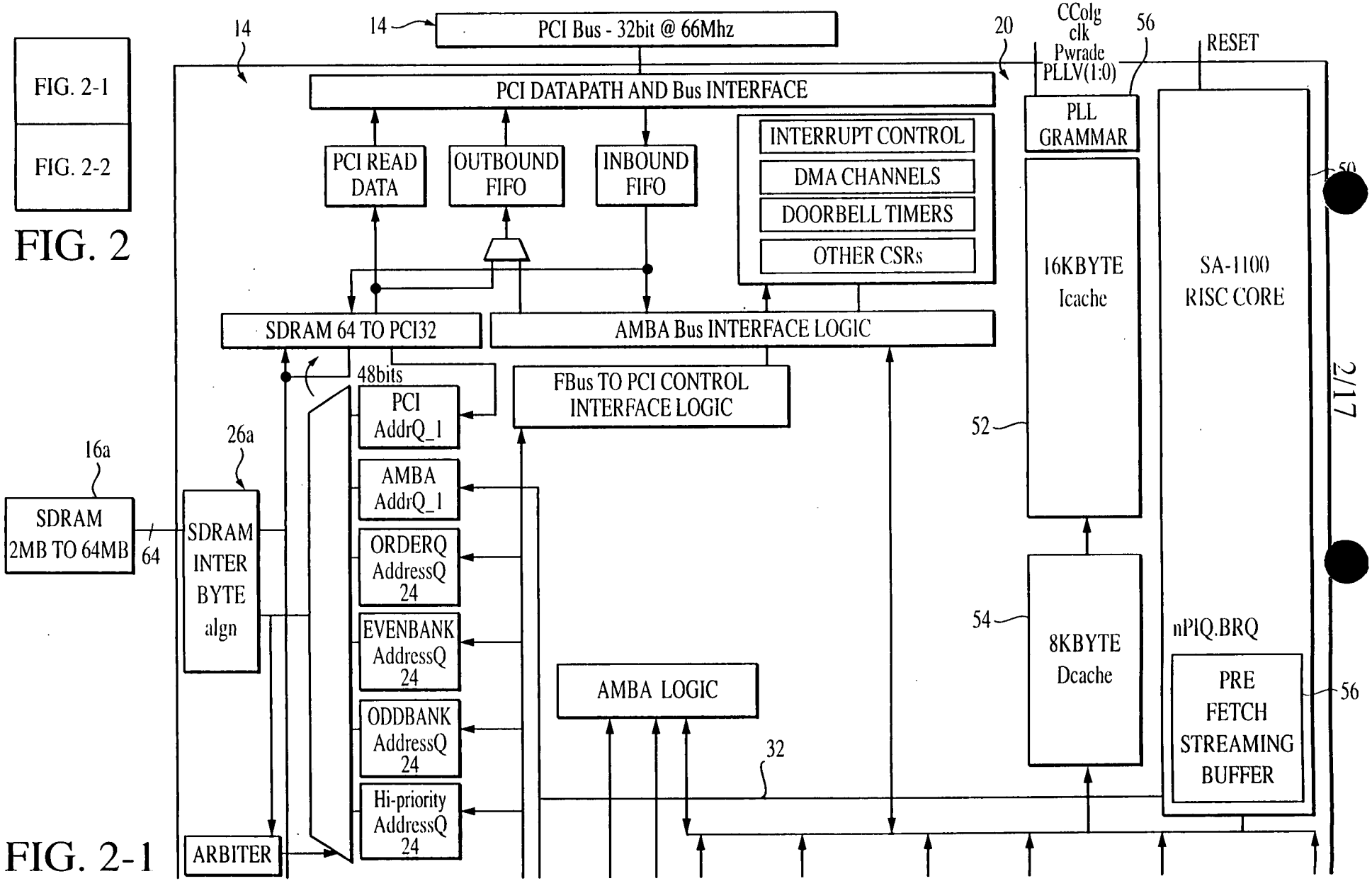


FIG. 2-1

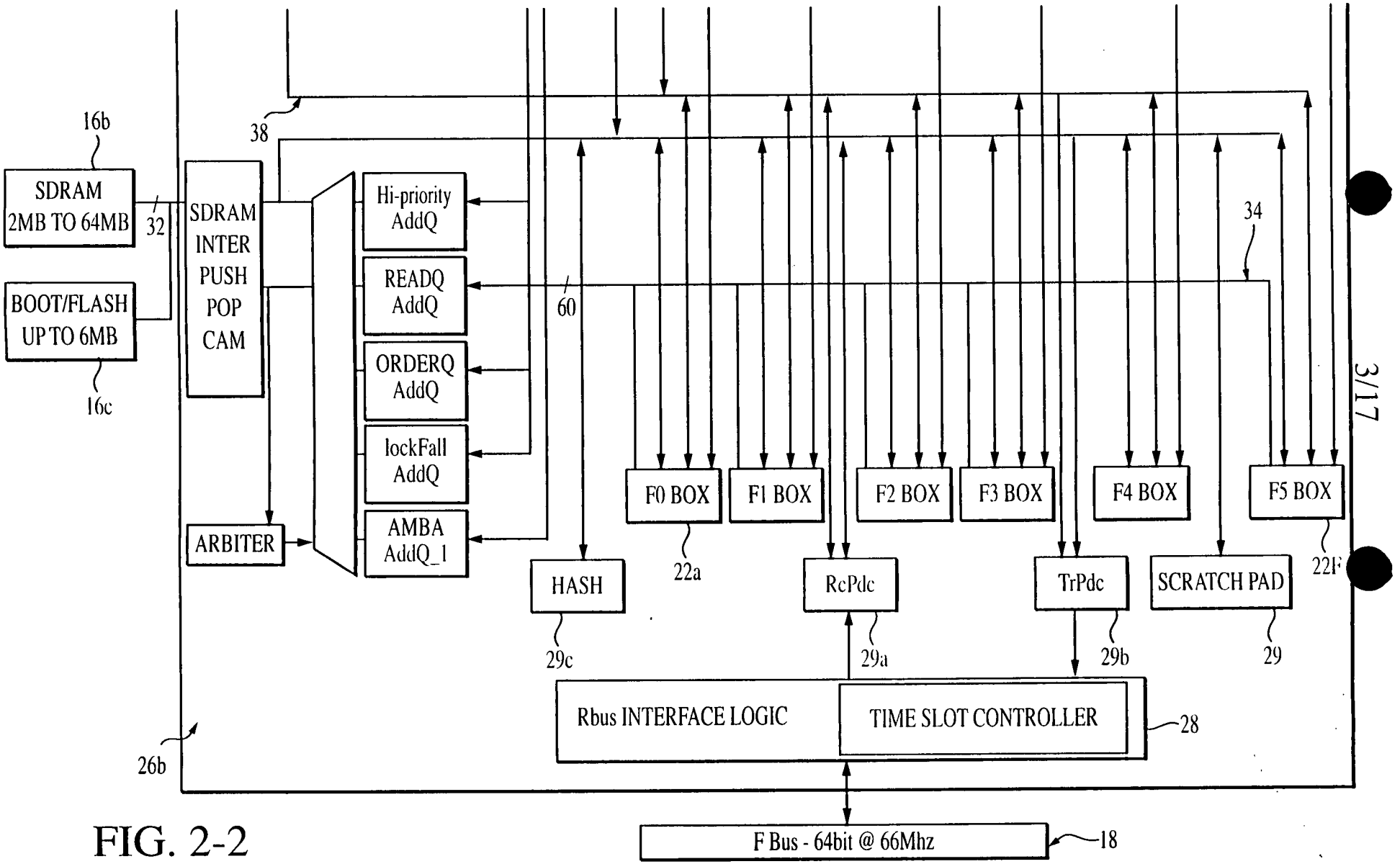

16b
SDRAM
2MB TO 64MB
32
BOOT/FLASH
UP TO 6MB
16c
SDRAM
INTER
PUSH
POP
CAM
38
Hi-priority
AddQ
READQ
AddQ
ORDERQ
AddQ
lockFall
AddQ
AMBA
AddQ_1
ARBITER
60
34
F0 BOX
F1 BOX
F2 BOX
F3 BOX
F4 BOX
F5 BOX
22a
22F
HASH
RcPdc
TrPdc
SCRATCH PAD
29c
29a
29b
29
Rbus INTERFACE LOGIC
TIME SLOT CONTROLLER
28
26b
F Bus - 64bit @ 66Mhz
18

FIG. 2-2

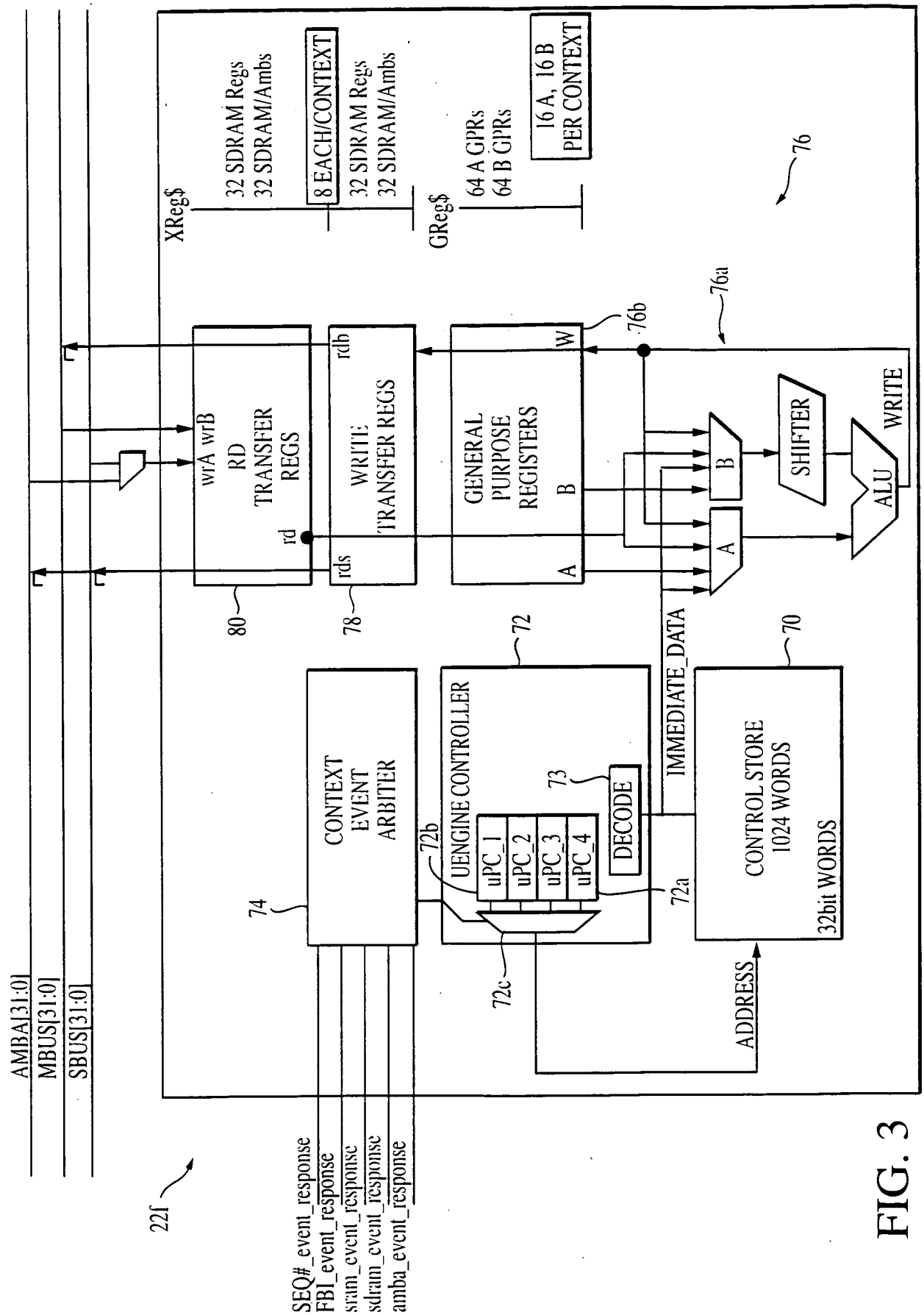

AMBA[31:0]
MBUS[31:0]
SBUS[31:0]
22f
SEQ#_event_response
FBI_event_response
sram_event_response
sdram_event_response
amba_event_response
XReg$
32 SDRAM Regs
32 SDRAM/Ambs
8 EACH/CONTEXT
32 SDRAM Regs
32 SDRAM/Ambs
GReg$
64 A GPRs
64 B GPRs
16 A, 16 B
PER CONTEXT
76
76a
76b
wrA wrB
RD
TRANSFER
REGS
rd
rdb
WRITE
TRANSFER REGS
rds
GENERAL
PURPOSE
REGISTERS
W
B
A
80
78
72
CONTEXT
EVENT
ARBITER
74
UENGINE CONTROLLER
72b
uPC_1
uPC_2
uPC_3
uPC_4
73
DECODE
72a
72c
IMMEDIATE_DATA
70
CONTROL STORE
1024 WORDS
32bit WORDS
ADDRESS
B
A
SHIFTER
ALU
WRITE

FIG. 3

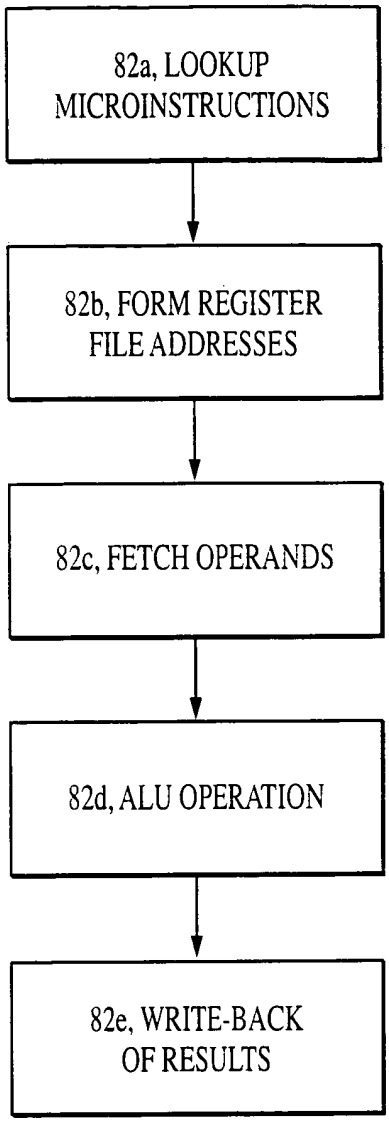
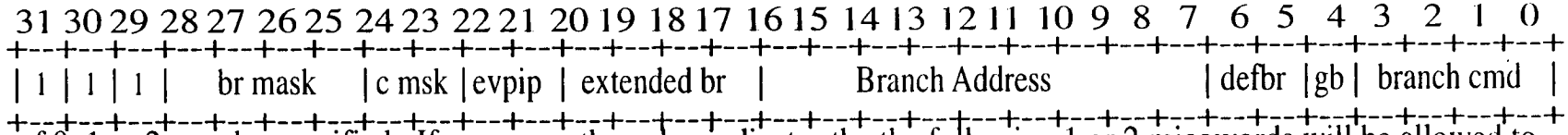

82a, LOOKUP MICROINSTRUCTIONS
82b, FORM REGISTER FILE ADDRESSES
82c, FETCH OPERANDS
82d, ALU OPERATION
82e, WRITE-BACK OF RESULTS

FIG. 4

branch instructions:



defbr: A value of 0, 1 or 2 may be specified. If non-zero, the value indicates the the following 1 or 2 micowords will be allowed to execute before the branch operation takes place.

gb: If set, guess that the branch path will be taken, thus prefetch the branch microword address. Otherwise prefetch the non-branch path. This field is only allowed to be set when defbr=0 or defbr=1.

branch address: branch address conditionally or unconditionally selected.

br_mask: Is decoded to the following options:

1) unconditional branch
2) branch when ALU<31>=1 (<0)
3) branch when ALU<31>=0 (>=0)
4) branch when ALU<31>=1 OR ALU<31:0>=0 (<=0)
5) branch when ALU<31>=0 AND ALU<31:0>!=0 (>0)
6) branch when ALU<31:0>=0 (=0)
7) branch when ALU<31:0>=1 (!=0)
8) branch when specified context mask = current context
9) branch when specified context mask ! = current context
10) branch on carry-out set
11) branch on carry-out clear
15) look at extended branch field to further decode branch type

extend_br: branches on various context-swapping signals or other signals

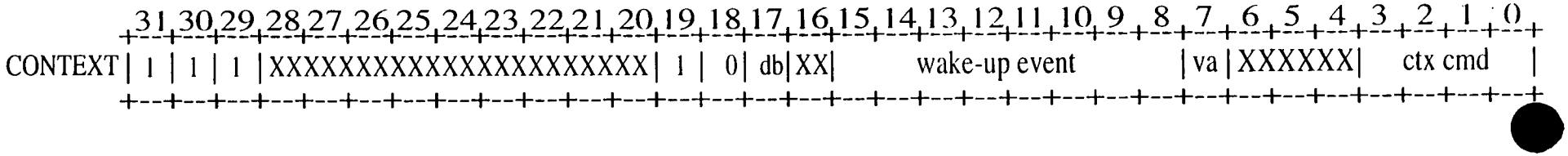
evpip: indicates pipe stage that this branch should be evaluated in

c msk: specifies a context number with which to conditionally branch on.

branch cmd: further specifies the type of branch, e.g., looks at condition codes of some other branch criteria

6/17

FIG. 5A



Context Descriptors:

1) Wake-up Events

0 = kill	8 = FBI	128 = SEQ_NUM_LSB
1 = voluntary	16 = INTER_THREAD	
2 = SRAM	32 = PCI_DMA_1	
4 = SDRAM	64 = PCI_DMA_2	

2) db → branch defer amount

3) va → value of sequence number

FIG. 5B

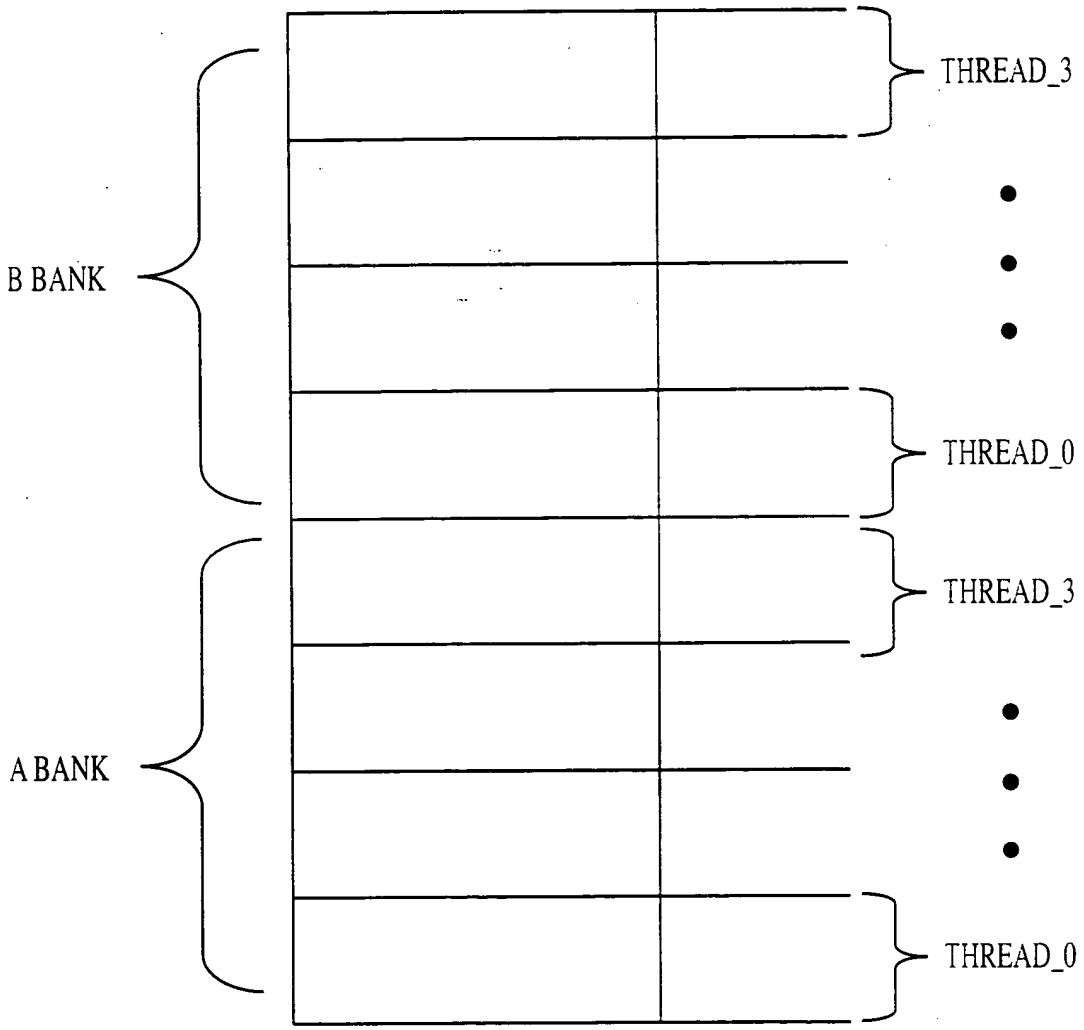


FIG. 6

FIG. 7-1

FIG. 7-2

FIG. 7

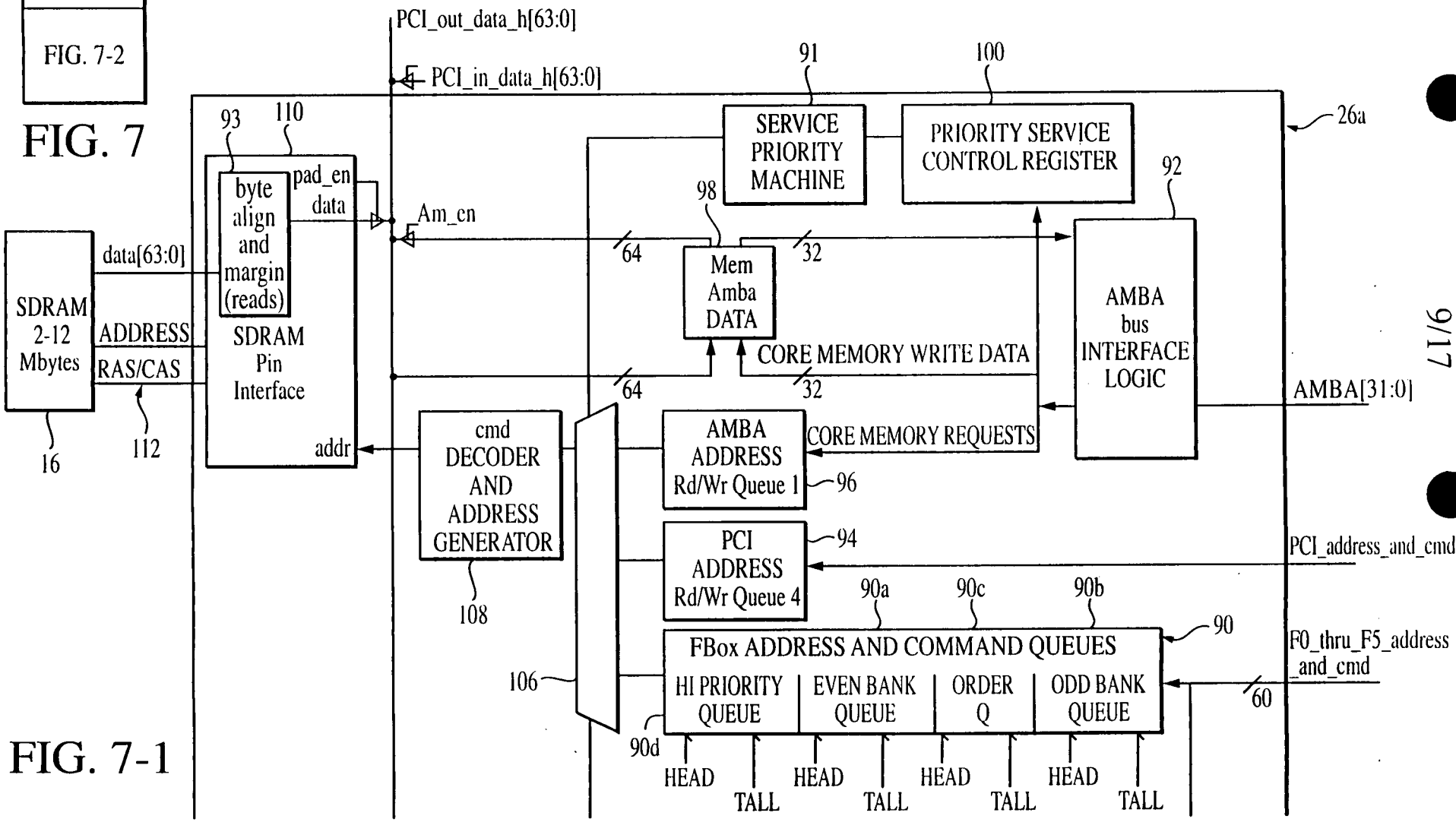


FIG. 7-1

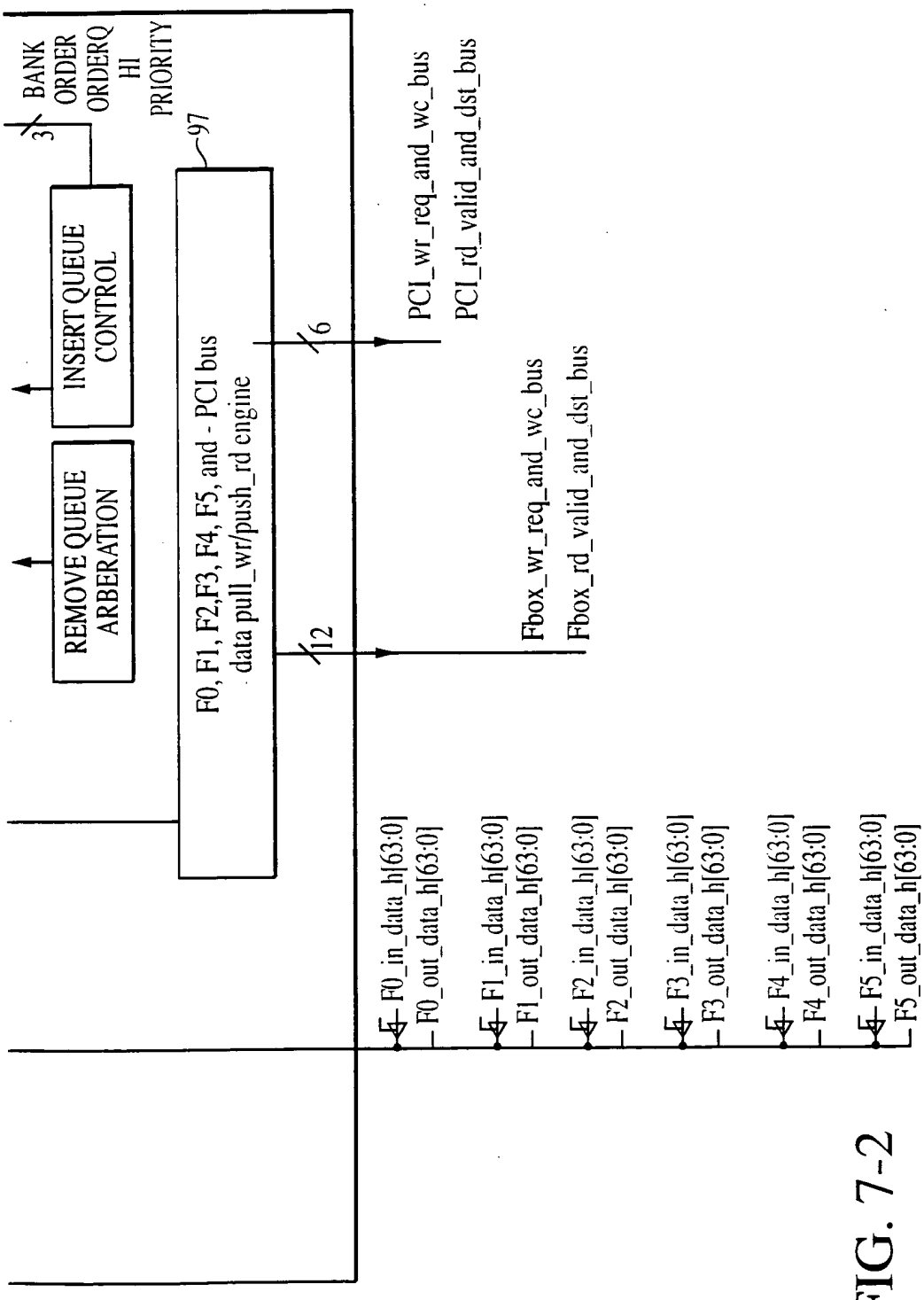

BANK ORDER ORDERQ HI PRIORITY
3
INSERT QUEUE CONTROL
REMOVE QUEUE ARBERATION
97
F0, F1, F2,F3, F4, F5, and - PCI bus data pull_wr/push_rd engine
6
12
PCI_wr_req_and_wc_bus
PCI_rd_valid_and_dst_bus
Fbox_wr_req_and_wc_bus
Fbox_rd_valid_and_dst_bus
F0_in_data_h[63:0]
F0_out_data_h[63:0]
F1_in_data_h[63:0]
F1_out_data_h[63:0]
F2_in_data_h[63:0]
F2_out_data_h[63:0]
F3_in_data_h[63:0]
F3_out_data_h[63:0]
F4_in_data_h[63:0]
F4_out_data_h[63:0]
F5_in_data_h[63:0]
F5_out_data_h[63:0]

FIG. 7-2

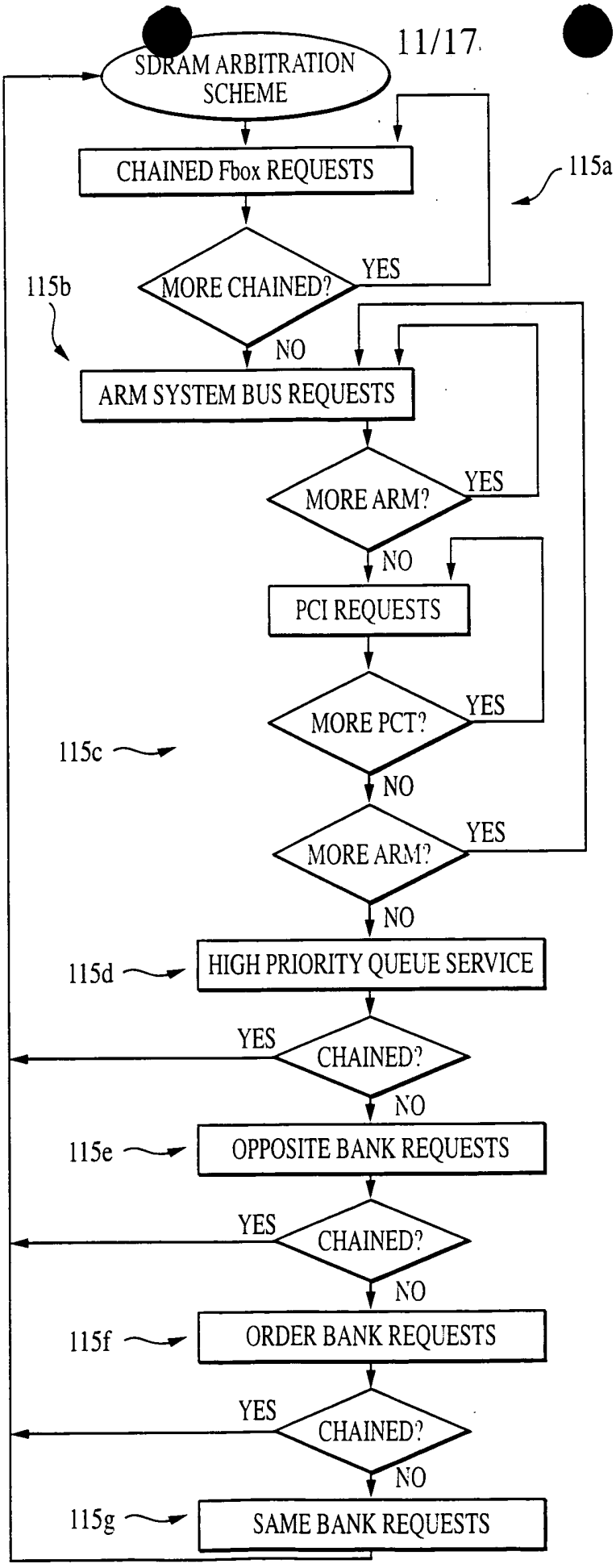
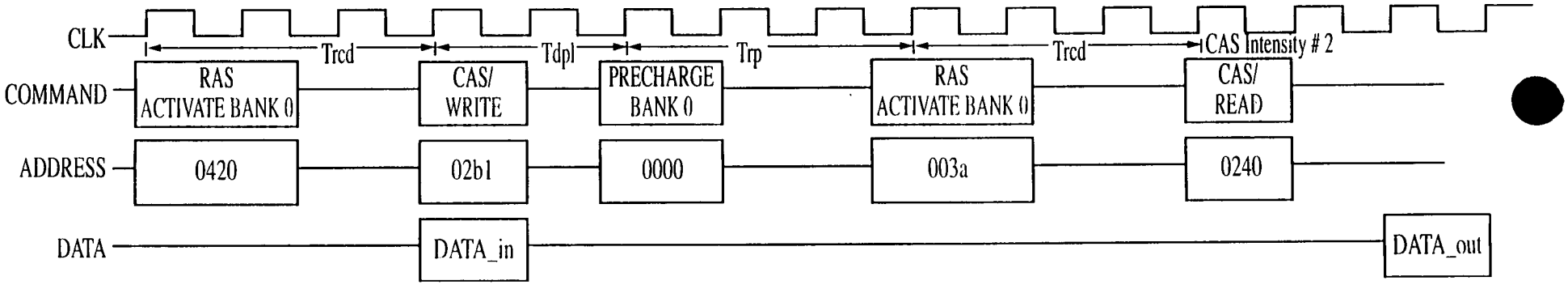

SDRAM ARBITRATION SCHEME
CHAINED Fbox REQUESTS
115a
MORE CHAINED?
YES
115b
NO
ARM SYSTEM BUS REQUESTS
MORE ARM?
YES
NO
PCI REQUESTS
MORE PCT?
YES
115c
NO
MORE ARM?
YES
NO
115d
HIGH PRIORITY QUEUE SERVICE
YES
CHAINED?
NO
115e
OPPOSITE BANK REQUESTS
YES
CHAINED?
NO
115f
ORDER BANK REQUESTS
YES
CHAINED?
NO
115g
SAME BANK REQUESTS

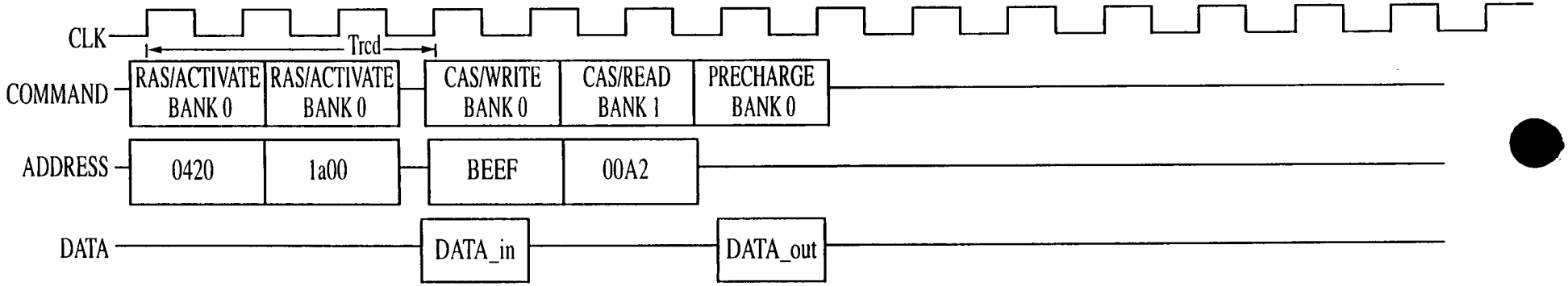
FIG. 7A

SINGLE QUADWORD WRITE FOLLOWED BY A SINGLE QUADWORD READ

WITHOUT ACTIVIE MEMORY OPTIMIZATION



WITH ACTIVIE MEMORY OPTIMIZATION



WHERE Trcd = RAS to CAS delay
Tdpl = DATA Input to Precharge Delay
Trp = Time to Precharge

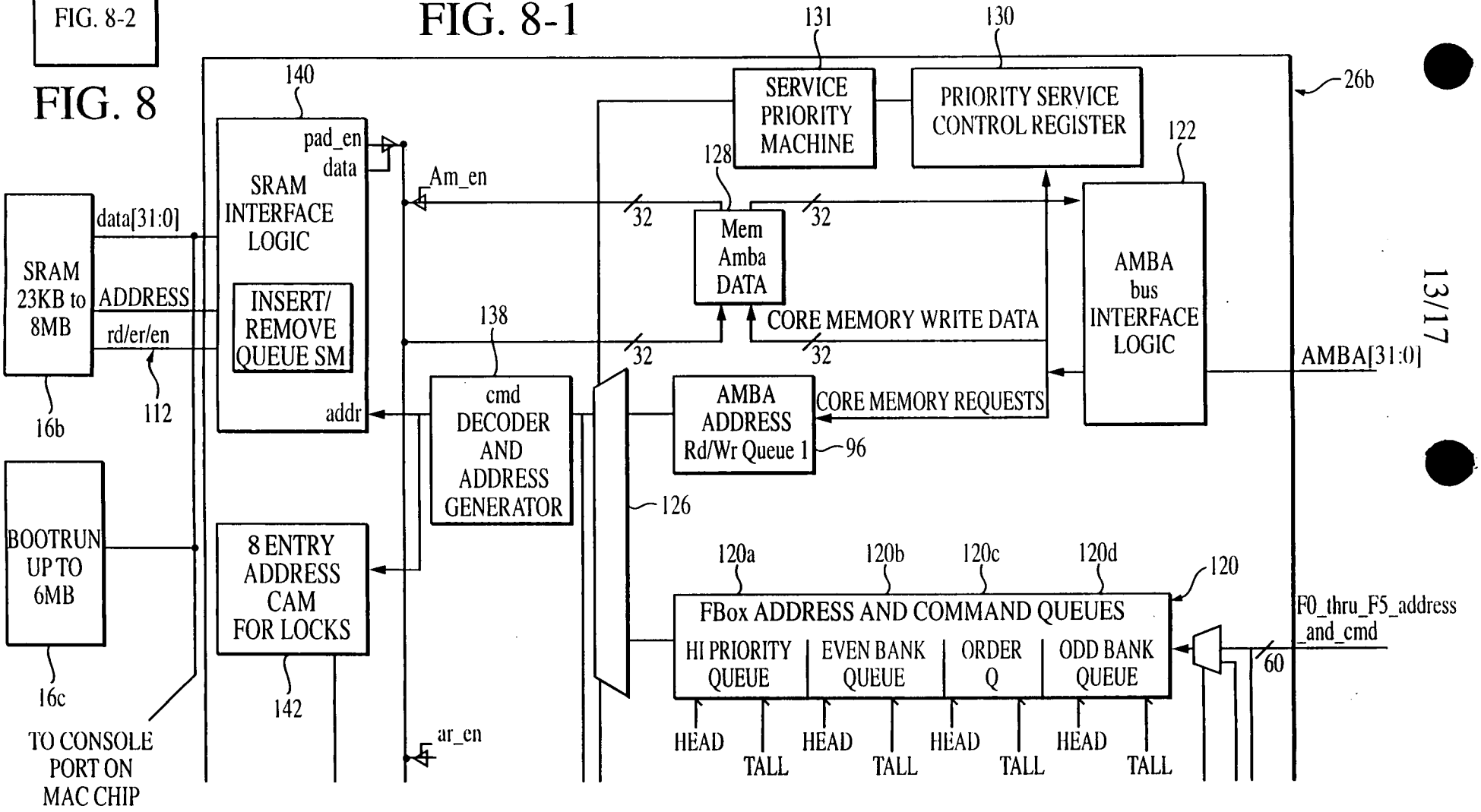
FIG. 7B

FIG. 8-1

FIG. 8-2

FIG. 8

FIG. 8-1



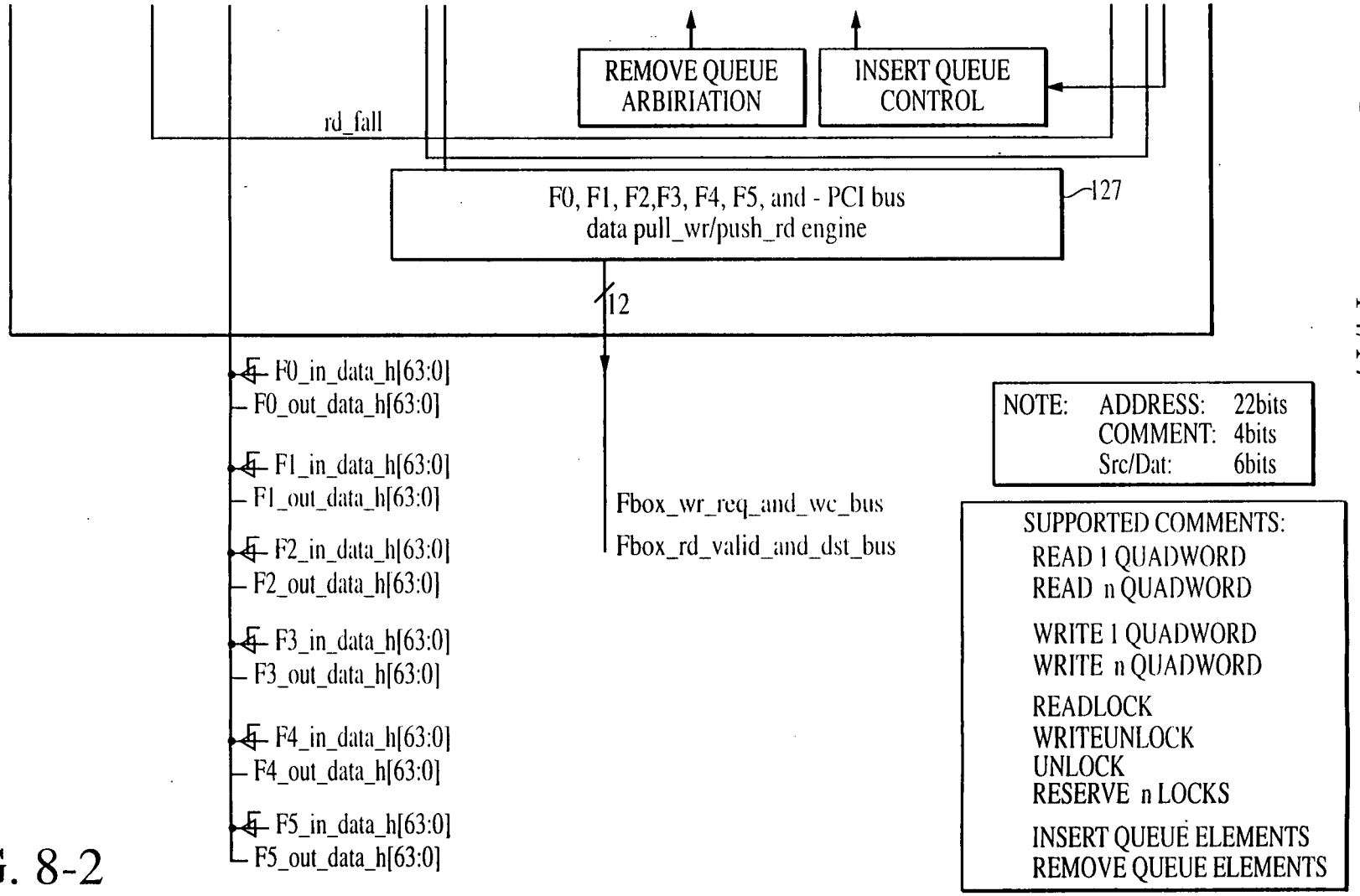
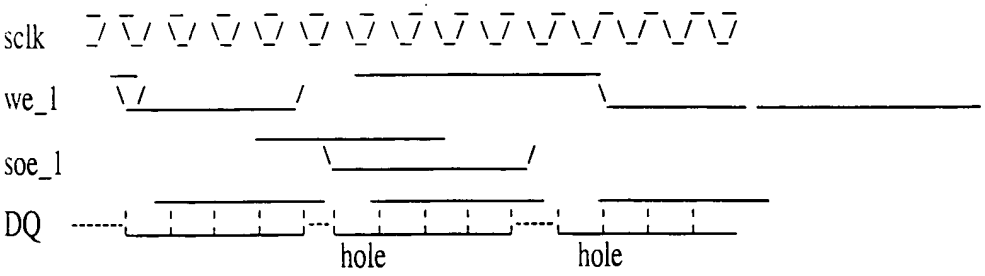
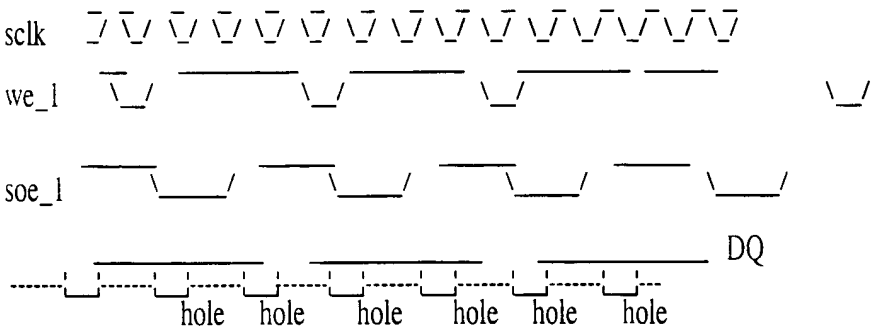

REMOVE QUEUE ARBIRIATION
INSERT QUEUE CONTROL
rd_fall
F0, F1, F2,F3, F4, F5, and - PCI bus data pull_wr/push_rd engine
127
12
F0_in_data_h[63:0]
F0_out_data_h[63:0]
F1_in_data_h[63:0]
F1_out_data_h[63:0]
F2_in_data_h[63:0]
F2_out_data_h[63:0]
F3_in_data_h[63:0]
F3_out_data_h[63:0]
F4_in_data_h[63:0]
F4_out_data_h[63:0]
F5_in_data_h[63:0]
F5_out_data_h[63:0]
Fbox_wr_req_and_wc_bus
Fbox_rd_valid_and_dst_bus
NOTE: ADDRESS: 22bits
COMMENT: 4bits
Src/Dat: 6bits
SUPPORTED COMMENTS:
READ 1 QUADWORD
READ n QUADWORD
WRITE 1 QUADWORD
WRITE n QUADWORD
READLOCK
WRITEUNLOCK
UNLOCK
RESERVE n LOCKS
INSERT QUEUE ELEMENTS
REMOVE QUEUE ELEMENTS

FIG. 8-2

4 WRITES AND 4 READS FOLLOWED BY MORE READS WITH OPTIMIZATION



4 WRITES AND 4 READS WITHOUT OPTIMIZATION



10 CYCLES VS. 14.

FIG. 8A

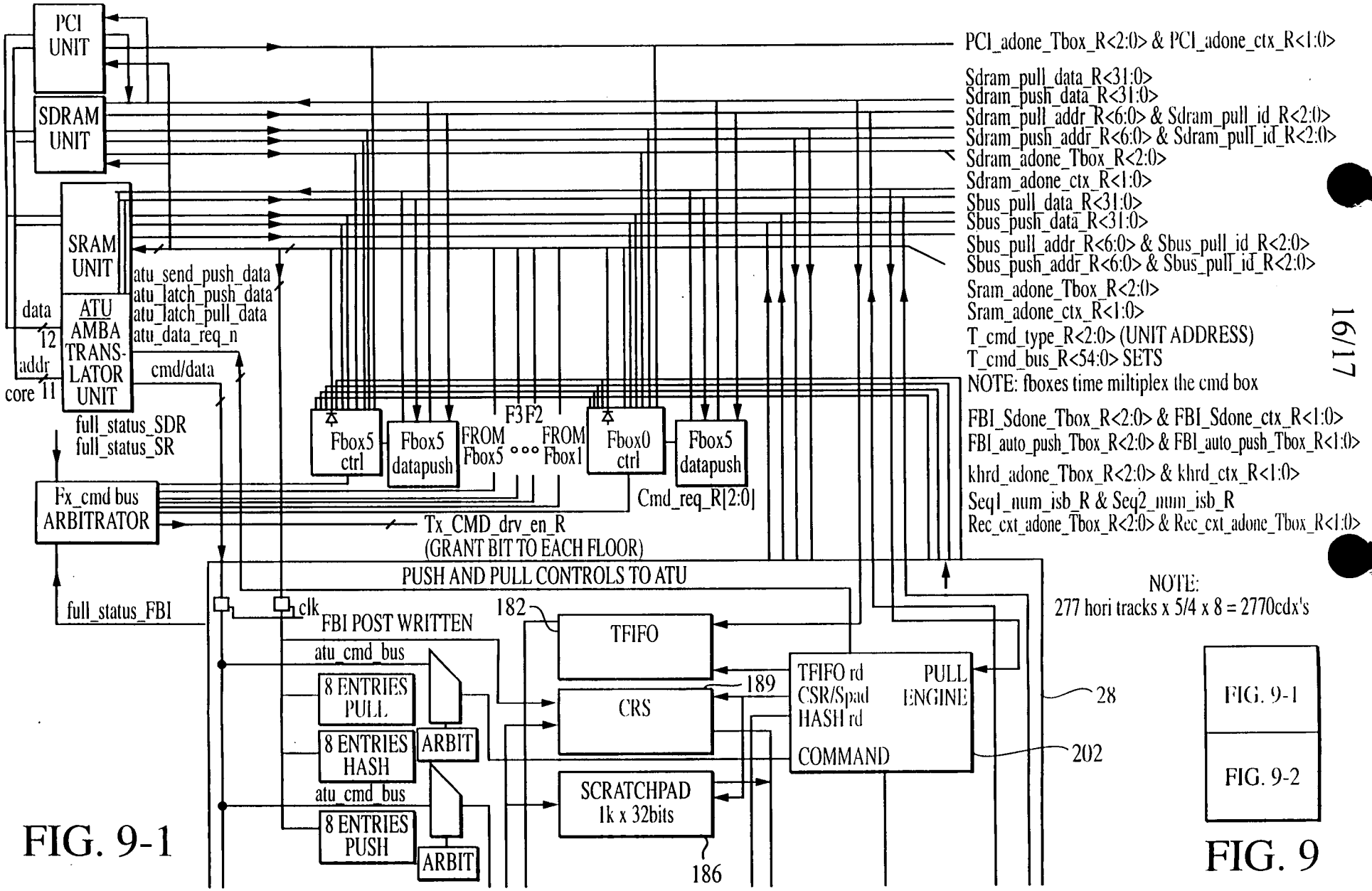
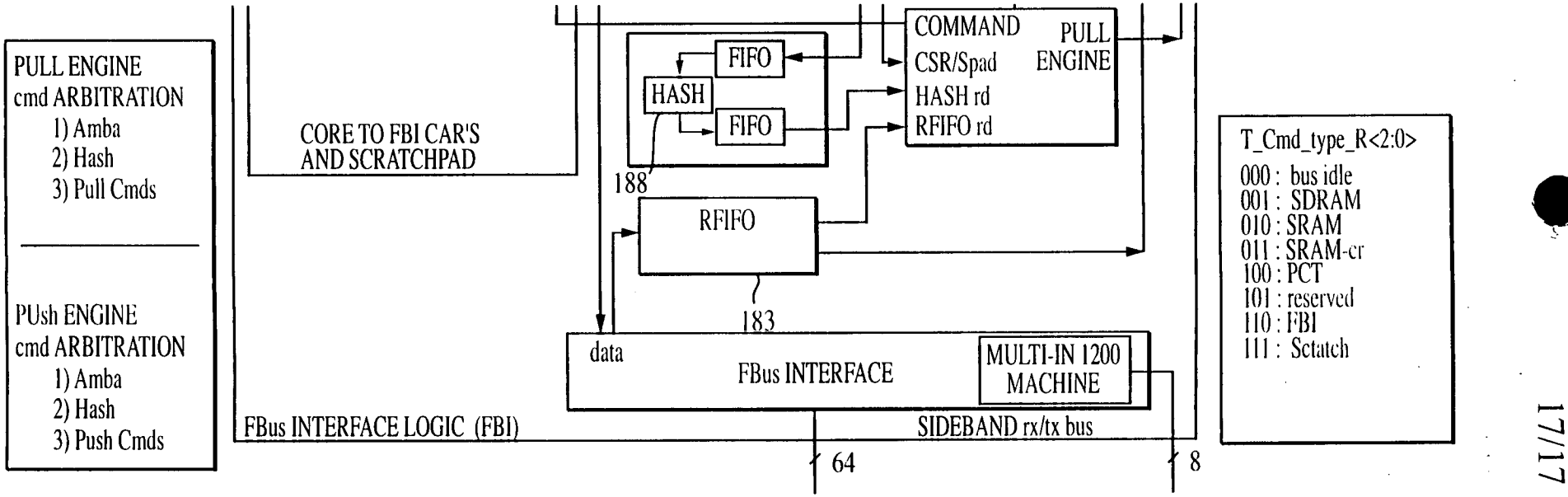

PCI UNIT
SDRAM UNIT
SRAM UNIT
ATU AMBA TRANS-LATOR UNIT
data
12
addr
core 11
atu_send_push_data
atu_latch_push_data
atu_latch_pull_data
atu_data_req_n
cmd/data
full_status_SDR
full_status_SR
Fx_cmd bus ARBITRATOR
full_status_FBI
Fbox5 ctrl
Fbox5 datapush
F3 F2
FROM Fbox5
FROM Fbox1
Fbox0 ctrl
Fbox5 datapush
Cmd_req_R[2:0]
Tx_CMD_drv_en_R
(GRANT BIT TO EACH FLOOR)
PUSH AND PULL CONTROLS TO ATU
clk
FBI POST WRITTEN
182
TFIFO
atu_cmd_bus
8 ENTRIES PULL
8 ENTRIES HASH
ARBIT
atu_cmd_bus
8 ENTRIES PUSH
ARBIT
189
CRS
SCRATCHPAD 1k x 32bits
186
TFIFO rd
CSR/Spad
HASH rd
COMMAND
PULL ENGINE
28
202
PCI_adone_Tbox_R<2:0> & PCI_adone_ctx_R<1:0>
Sdram_pull_data_R<31:0>
Sdram_push_data_R<31:0>
Sdram_pull_addr_R<6:0> & Sdram_pull_id_R<2:0>
Sdram_push_addr_R<6:0> & Sdram_pull_id_R<2:0>
Sdram_adone_Tbox_R<2:0>
Sdram_adone_ctx_R<1:0>
Sbus_pull_data_R<31:0>
Sbus_push_data_R<31:0>
Sbus_pull_addr_R<6:0> & Sbus_pull_id_R<2:0>
Sbus_push_addr_R<6:0> & Sbus_pull_id_R<2:0>
Sram_adone_Tbox_R<2:0>
Sram_adone_ctx_R<1:0>
T_cmd_type_R<2:0> (UNIT ADDRESS)
T_cmd_bus_R<54:0> SETS
NOTE: fboxes time miltiplex the cmd box
FBI_Sdone_Tbox_R<2:0> & FBI_Sdone_ctx_R<1:0>
FBI_auto_push_Tbox_R<2:0> & FBI_auto_push_Tbox_R<1:0>
khrd_adone_Tbox_R<2:0> & khrd_ctx_R<1:0>
Seq1_num_isb_R & Seq2_num_isb_R
Rec_cxt_adone_Tbox_R<2:0> & Rec_cxt_adone_Tbox_R<1:0>
NOTE:
277 hori tracks x 5/4 x 8 = 2770cdx's

FIG. 9-1

FIG. 9-1
FIG. 9-2

FIG. 9



17/17

ATU NOTES:
a) CORE TO FBoxRegs:
USE aram_push_data_bus
b) CORE TO FBI Regs:
USE PRIVATE ATU/FBI
cmd/data bus
c) CORE READS FBoxRegs:
USE Sram_pull_data_bus
d) CORE READS FBI Regs:
USE sram_push_data_bus
(UNLESS sram APPEAR FAKE
ANOTHER Fbox To FBI ON
sram_push_bus)

Card_Req_R,2:0>
000 NONE
001 Sram Chain
010 SDR Chain
011 Sram
100 SDR
101 FBI
110 PCI
111

Tx_CMD_drv_en_R<1:0>
0 NONE
1 GRANT

Sdram_puXX_addr_R<6:0>
[4:0] xfer_reg_addr
IF NOT TFIFO
[6:0] TIFIFO_addr

Sdram_puXX_ID_R<6:0>
0 - 5 Fboxes
8 - 13 Fboxes-csr
6 fbl
15 nop

Sdram_puXX_addr_R<6:0>
[4:0] xfer_reg_addr

Sdram_puXX_ID_R<6:0>
0 - 5 Fboxes
8 - 13 Fboxes-csr
6 fbl
15 nop

Fbox BRANCH/Ctx CHOICES

1) FBI_adone		br / ctx
2) FBI_auto_push		br / ctx
3) Ithread_adone		br / ctx
4) signal_rec_cxt		br / ctx
5) Seq#1_change	(flag)	br / ctx
6) Seq#2_change	(flag)	br / ctx
7) SRAM_adone		br / ctx
8) SDRAM_adone		br / ctx
9) volunteer_ctx_swap		ctx
10) Req_req_avaliable	(flag)	br
11) SDRAM rd parity err		br
12) Fbox_push_protect	(flag)	br
13) ccodes, contexts and kill		

FIG. 9-2